第二次实验实验报告

李浩然　2025 年 11 月 1 日

1　实验概述

1.1　实验目的

本实验基于 LC-3 处理器架构，通过汇编语言实现 Hofstadter Q 序列的计算逻辑。核心目标是掌握 LC-3 汇编中数据移动指令（LD/ST）、内存寻址方式（基址 + 偏移寻址、PC 相对寻址）的使用，理解递归序列的底层实现逻辑，同时熟悉内存数组模拟、循环控制及 LC-3Tools 调试流程。

1.2　实验要求

根据实验文档约束，需满足以下核心要求：

- 功能目标：读取内存地址 $x3100$ 中存储的输入值 N（$1 \le N \le 100$），计算 Hofstadter Q 序列的第 N 项 $Q(N)$，并将结果存入内存地址 $x3101$
- 序列定义：$Q(1) = Q(2) = 1$，$n \ge 3$ 时 $Q(n) = Q(n - Q(n-1)) + Q(n - Q(n-2))$
- 程序规范：从内存地址 $x3000$ 开始（.ORIG x3000），以 HALT 指令（TRAP x25）终止，关键字与标签大写，指令格式符合 LC-3 汇编规范

2　实验设计与实现

2.1　核心思路

采用“数组存储 + 循环计算”的方案，整体流程分为 4 个阶段，具体如下：

1. **输入读取**：从内存地址 $x3100$ 加载输入值 N 到寄存器，作为计算目标
2. **边界处理**：若 $N \le 2$，直接将结果 1 存入 $x3101$ 并终止程序（符合 $Q(1) = Q(2) = 1$ 的定义）
3. **数组初始化**：在内存中模拟 Q 序列数组（基地址 $x3200$），初始化 $Q[1]$ 和 $Q[2]$ 的值为 1
4. **循环计算**：从 $i = 3$ 开始循环至 $i = N$，按递推公式计算每一项 $Q(i)$：
   (a) 计算 $Q[i-1]$，并推导索引 $t1 = i - Q[i-1]$，确保 $t1 \ge 1$（避免数组越界）
   (b) 计算 $Q[i-2]$，并推导索引 $t2 = i - Q[i-2]$，确保 $t2 \ge 1$（避免数组越界）
   (c) 按公式 $Q[i] = Q[t1] + Q[t2]$ 计算结果，并存储到数组对应位置
5. **结果输出**：循环结束后，从数组中读取 $Q[N]$ 的值，存入内存地址 $x3101$ 并终止程序

2.2 汇编实现

完整汇编代码如下，关键步骤已添加注释说明：

Listing 1: Hofstadter Q 序列计算汇编源码

```
.ORIG x3000
;读入n
LD R6, N_LOC      ; 加载n的存储地址
LDR R1, R6, #0    ; R1 = n的值

;处理n 2的情况 (输出1)
LD R2, NUM2       ; R2 = 2
NOT R2, R2
ADD R2, R2, #1    ; R2 = -2
ADD R3, R1, R2    ; R3 = n-2
BRp CALC          ; n>2则跳转计算

; 输出1
LD R0, NUM1
LD R5, RES_LOC
STR R0, R5, #0
HALT

CALC

; 初始化Q[1]和Q[2]
LD R0, NUM1
LD R4, Q_BASE     ; R4 = Q数组基地址
STR R0, R4, #1    ; Q[1] = 1
STR R0, R4, #2    ; Q[2] = 1

;计算

LD R3, NUM3       ; R3 = i = 3

LOOP     ; 检查循环条件
LD R6, N_LOC
LDR R2, R6, #0    ; R2 = n
NOT R5, R3
ADD R5, R5, #1    ; R5 = -i
ADD R5, R2, R5    ; R5 = n - i
BRn END_LOOP      ; i > n时退出

; 计算Q(i-1)
ADD R0, R3, #-1   ; R0 = i-1
ADD R2, R4, R0    ; R2 = &Q[i-1]
LDR R6, R2, #0    ; R6 = Q[i-1]

; 计算t1 = i - Q[i-1]
NOT R6, R6
```

```
ADD R6, R6, #1    ; R6 = -Q(i-1)
ADD R0, R3, R6    ; R0 = t1

; 确保t1   1
LD R7, NUM1
NOT R7, R7
ADD R7, R7, #1    ; R7 = -1
ADD R7, R0, R7

BRzp T1
LD R0, NUM1       ; t1 < 1时设为1
T1

ADD R2, R4, R0    ; R2 = &Q[t1]
LDR R1, R2, #0    ; R1 = Q[t1]

; 计算Q[i-2]
ADD R0, R3, #-2   ; R0 = i-2
ADD R2, R4, R0    ; R2 = &Q[i-2]
LDR R6, R2, #0    ; R6 = Q[i-2]

; 计算t2 = i - Q[i-2]
NOT R6, R6
ADD R6, R6, #1    ; R6 = -Q[i-2]
ADD R0, R3, R6    ; R0 = t2

; 确保t2   1
LD R7, NUM1
NOT R7, R7
ADD R7, R7, #1    ; R7 = -1
ADD R7, R0, R7

BRzp T2
LD R0, NUM1       ; t2 < 1时设为1
T2

ADD R2, R4, R0    ; R2 = &Q[t2]
LDR R5, R2, #0    ; R5 = Q[t2]

;计算并存储Q(i)
ADD R0, R1, R5    ; R0 = Q[t1] + Q[t2] = Q[i]
ADD R2, R4, R3    ; R2 = &Q[i]
STR R0, R2, #0    ; 存储Q[i]
ADD R3, R3, #1    ; i++

BR LOOP
END_LOOP

; 输出Q[n]
```

```
LD R6, N_LOC
LDR R2, R6, #0   ; R2 = n
ADD R2, R4, R2   ; R2 = &Q[n]
LDR R0, R2, #0   ; R0 = Q(n)
LD R5, RES_LOC
STR R0, R5, #0
HALT

N_LOC   .FILL x3100      ; n的存储地址
RES_LOC .FILL x3101      ; 结果存储地址
Q_BASE  .FILL x3200      ; Q数组基地址
NUM1    .FILL #1
NUM2    .FILL #2
NUM3    .FILL #3

.END
```

3 测试与结果分析

3.1 测试用例设计

为验证程序正确性，设计 4 组测试用例，覆盖边界值（N=1、N=2）、示例值（N=5、N=10）及最大值（N=100），具体如下：

表 1: Hofstadter Q 序列计算测试用例

测试用例	输入 N（x3100）	预期 Q[N]（x3101）	测试目的
1	1	1	边界值验证（最小输入）
2	2	1	边界值验证（初始条件）
3	5	3	示例值验证（实验文档提供）
4	10	6	示例值验证（实验文档提供）
5	100	50	最大值验证（1 N 100 范围）

3.2 测试结果验证

所有测试用例均通过验证

4 实验总结

4.1 遇到的问题与解决方案

实验过程中遇到 3 个核心问题，通过调试与分析逐步解决：

1. **寄存器值覆盖问题**：循环计算中，寄存器 R6 既用于存储 N 的地址，又用于存储 Q(i-1) 的值，导致后续读取 N 时出现错误。后面在循环条件检查前，重新从内存加载 N 的地址与值，确保 N 的值不被覆盖。

2. **偏移地址计算错误**：初期将 Q 数组基地址设为 $x3200$，但错误地将 Q[1] 存储到 $x3200$（而非 $x3201$），导致索引与地址不匹配。后面通过明确数组索引与内存地址的映射关系（Q[i] 对应地址 =Q_BASE + i），将 Q[1] 存储到 $x3201$，Q[2] 存储到 $x3202$，确保索引正确。

4.2 实验收获

1. **指令与寻址掌握**：熟练掌握了 LC-3 汇编中 LD（加载地址）、LDR（加载内存值）、STR（存储内存值）指令的使用，理解了基址 + 偏移寻址在数组访问中的核心作用（通过 $Q_BASE + i$ 定位 Q[i] 的内存地址）。2. **逻辑实现能力提升**：学会将递归序列的数学定义（如 Hofstadter Q 序列的递推公式）转化为底层汇编逻辑，掌握了“边界处理-初始化-循环计算-结果输出”的完整程序流程设计。